

Does Extra Genetic Diversity Maintain Escalation in a Co-Evolutionary Arms Race

Paul J. Darwen

Cognitive Science Research Group
Dept. of Computer Science and Electrical Engineering
The University of Queensland
Brisbane, Queensland 4072
Australia
Email: darwen@csee.uq.edu.au

Xin Yao

EEBIC Group
School of Computer Science
The University of Birmingham
Edgbaston, Birmingham B15 2TT
United Kingdom
Email: x.yao@cs.bham.ac.uk

Abstract-

In evolutionary computation (EC), genetic diversity (or its absence) gets the credit (or the blame) for a multitude of effects — and so mutation operators, population initialization, and even pseudo-random number generators, all get probed and prodded to improve genetic diversity. This paper demonstrates how extra initial diversity can appear to cause improvements in the performance of co-evolutionary learning, but the true cause is in unforeseen effects of the problem-specific representation. The learning task considered in this paper is a variation on the game of Iterated Prisoner’s Dilemma (IPD): here players have a fine-grained range of intermediate choices between full cooperation and full defection.

1 Introduction

1.1 Diversity: Witch Hunt or Panacea

In evolutionary computation (EC), genetic diversity (or the lack thereof) gets the credit (or blame) for numerous effects, often without testing for other causes. There is a widespread faith that genetic diversity is always a good thing in EC, in all possible applications and under all possible circumstances.

This dogma has driven much EC research about how to increase or maintain genetic diversity: mutation operators [16], population initialization [12], and even pseudo-random number generators [21] have all been studied with the unstated assumption that many problems could be solved if only genetic diversity could somehow be increased or maintained. Genetic Programming takes diversity so seriously, it has the specialized “ramped half-and-half” method of initializing the population [17, pages 91-94].

These studies have generally been done without looking for any unforeseen interactions between genetic diversity and the particular implementation of the sample tasks. This blatant disregard for the idiosyncrasies peculiar to the chosen problem, and the unseemly haste in blaming genetic diversity as the *cause* for everything that merely *correlates* with genetic diversity, is what this paper will rebut.

As a sample task, this paper uses a *co-evolutionary* genetic algorithm (described more fully in Sections 1.2 and 2.1)

to learn the Iterated Prisoner’s Dilemma (IPD) game, following Axelrod [2]. Instead of players having merely two choices of action (full cooperation or full defection), here and in other studies [13] [24] [27] players have a more fine-grained choice of actions, ranging between full cooperation and full defection. This more closely resembles many real-world situations [24, page 175].

1.2 Co-Evolutionary Learning: Why Stagnate There?

In evolutionary computation (EC), a population of trial solutions is evaluated by a *fitness function*, usually hand-written by a human expert. In contrast, *co-evolution* is where the fitness of a trial solution depends on the other trial solutions in the current population. For example, in a population of strategies for a game, the fitness of each strategy is how many wins it achieves against the other strategies in the same population.

This approach has worked well on numerous games (including Checkers (Draughts) [7] and Backgammon [23] [8]) and other tasks that resemble games in certain ways, such as schedule optimization [15] and robotics [18].

Co-evolutionary learning sets up an escalating “arms race” of innovation. Starting from a random initial population of strategies, co-evolution can learn to play a game, without prior knowledge from human experts on that game.

However, this upwards spiral eventually stagnates at a certain level of ability. Exactly why it should stagnate at the level it does, instead of at a higher level, is not understood. Ideally, it should keep escalating in ability until it exceeds human ability on that task.

Given that the learning environment in co-evolution is, in fact, the other members of the evolving population, one possible way to keep the “arms race” moving upwards is to encourage a population with more genetic diversity: a more varied population might mean a more varied co-evolutionary learning environment, and this might enable better and more robust learning. That possibility is addressed here.

This paper considers the effect of varying the initial diversity in a co-evolving population of strategies for the game of Iterated Prisoner’s Dilemma (IPD).

1.3 Overview

Section 2 discusses in more detail the issues of co-evolutionary learning and its stagnation, as well as the idiosyncrasies of two popular learning tasks, the games of Backgammon and Iterated Prisoner’s Dilemma (IPD).

Section 3 describes how the co-evolutionary genetic algorithm is implemented, to learn IPD. It is an utterly conventional approach. Each IPD strategy is a simple feed-forward neural network, represented as a fixed-length string of floating-point numbers. Each player in the current population plays against every other player (including itself), and its fitness is its total score from those games.

Results are in Section 4. In Section 4.2, increased genetic diversity in the initial population *appears* to make mutual cooperation more likely. It looks convincing that diversity encourages the “evolution of cooperation”.

But is diversity really the cause? Alas, no. Section 4.3 shows that the ultimate cause is not genetic diversity as such, but a perverse interaction with the problem-specific implementation. In brief, more initial genetic diversity reduces the behavioral diversity. It skews the decisions made, so that the players are not actually learning the complicated IPD with fine-grained choices of cooperative level, but the classic and greatly simplified IPD with only two choices. Too much genetic diversity is actually *worse* for learning this particular task.

Section 5 discusses these results, and the paper concludes in Section 6.

2 Background

2.1 Co-Evolutionary Learning

Evolutionary computation maintains a population of trial solutions for the problem to be solved. An *evaluation function* measures the quality of each trial solution. Usually, a human expert writes that evaluation function, knowing in advance how to accurately evaluate a trial solution.

This usual approach is ill-suited for learning to play a game of conflict, for two reasons.

First, for many game-like applications, human experts are simply unavailable. This is often the case for military tasks, where new weapons take time to learn to use. Human expertise is also unavailable for investment management; even if one could afford to hire some Armani-wearing Porsche-driving human traders to write software, the vast majority of mutual funds actually perform *worse* than the market average [11] [20]. Also, new markets — such as recently privatized electricity markets in California, Queensland, and the United Kingdom, as well as new stock exchanges in developing nations — are so new that they lack a history for any human to have learned from, so all humans are novices. For such tasks, human experts are not available to write a fitness function.

Second and more importantly, in most game-like tasks, being able to accurately evaluate a strategy often requires the

same knowledge on how to create a good strategy in the first place. That is, many tasks are such that if a human can judge a strategy, the same human can often hand-write a reasonable strategy anyway, thus avoiding any need for machine learning. To put it another way, if the famous artist Picasso was available, and you wanted a nice work of art, would you rather have Picasso write a fitness function for evolving art works, or instead just let him paint paintings directly?

The interesting question is: how to evolve a good solution even if you *don’t* have a resident Picasso? If you have a Picasso, art will result one way or another, but how can a machine exceed human ability?

Co-evolution is one possible way to create an expert-level strategy without a human expert; a way of exceeding the ability of its creator (or, potentially, of any living person) to create super-human skill for a particular task.

Co-evolution is where each trial solution in an evolving population is evaluated by its peers in the same population (or perhaps by another population evolving in tandem [14]). The evaluation function is the rest of the population, which changes as it evolves. The aim is to set up an escalating arms race of innovation.

Co-evolution’s most popular application is to learn to play a board game, such as Checkers [7] and Backgammon [23] [8]. Each individual in the population is a strategy, and plays that game against every other individual in the current population (or perhaps another population evolving in tandem [14]). A strategy’s fitness is the average score of all the games it plays against the current generation. These are usually 2-player (i.e., pairwise) interactions, but they can also be n -player interactions [26].

Co-evolution also works well on certain non-game tasks, such as co-evolving a population of sorting *algorithms* against a population of sorting *problems* [14].

Starting from a random initial population, co-evolution requires no prior knowledge of how to play the game well. As the population evolves, the current population contains improved strategies, and thus becomes a more accurate testing ground for its members. The fitness function is thus a moving target, as the population becomes better at playing the game. This increasing level of difficulty in the co-evolutionary evaluation function causes an escalation in the ability of the fittest survivors.

2.1.1 Co-Evolution and Stagnation

A typical attempt at co-evolutionary learning results in performance like that of Figure 1: from a random initial population, and playing only strategies within the evolving population, co-evolution learns to an impressive level of ability [8] compared to an unseen benchmark Backgammon player¹, after learning is completed — co-evolution does not learn from any external trainer.

¹The benchmark used here is Pubeval, generously made available by Gerald Tesauro of IBM. The comparison in Figure 1 follows the Pub-1 column in [25, page 242].

But eventually, learning in Figure 1 stagnates at a plateau. It doesn't improve beyond a certain point.

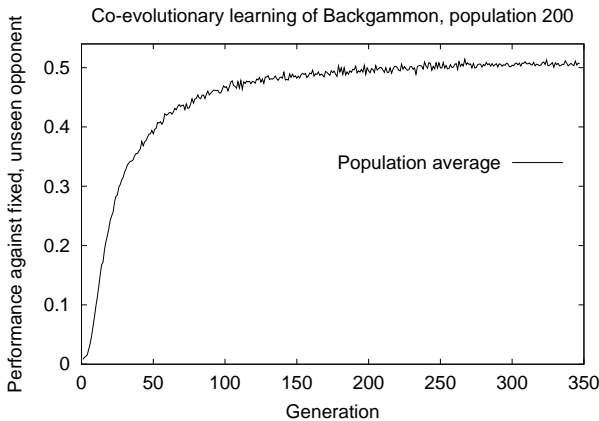


Figure 1: Typical co-evolutionary learning [8]. All 200 strategies in the population play Backgammon against all the other 199 strategies. Fitness is the total number of wins. Compared to an unseen benchmark Backgammon strategy called Pubeval [25, page 242], co-evolution learns to play well, but eventually stagnates and stops improving [8].

Co-evolution attempts to induce an upward-spiraling “arms race” of increasing sophistication and ability, so why does it stagnate at the level it does? In Figure 1, it stagnates slightly ahead of the unseen benchmark; why didn't it stagnate further up, or further down? How can we keep escalation going up further, so that it tops out at a level of ability greater than human ability?

There is no clear-cut understanding of what determines the peak level of a co-evolutionary arms race. One suggestion is that tasks with randomness (such as Backgammon, which has dice) are especially suited to keeping the arms race rising [23]. Section 2.1.2 discusses this.

2.1.2 Stochastic Learning Tasks

Pollack and Blair [23, page 233] suggest that the success of a co-evolutionary learning system may depend more on the attributes of the task, than on the details of the particular learning algorithm used. In particular, the game of Backgammon has the randomizing element of dice: this (they suggest) forces the game in varied and unpredictable directions. A broad range of scenarios encourages robust learning.

To underline their case, Pollack and Blair [23] demonstrated that co-evolution on Backgammon works surprisingly well with a population size of only 2 — i.e., basically a hill-climber. This agrees with their argument that randomness in the learning task makes co-evolution easier. More elaborate algorithms — such as temporal difference learning [25] and co-evolution with a larger population [8] — learn Backgammon better than simple hill-climbing, but not much better.

Not all learning tasks, however, resemble Backgammon. Many tasks lack a source of randomness such as Backgam-

mon's dice. For example, co-evolutionary learning is often applied to the abstract mathematical game of Iterated Prisoner's Dilemma (IPD), which is usually implemented without randomness or noise (but not always [3]). Despite IPD's lack of random dice, co-evolution nonetheless works quite well in discovering good strategies [2] [10].

Also, Backgammon (like any zero-sum game with perfect information) necessarily offers a perfect minimax strategy. IPD is non-zero-sum and lacks such a single perfect strategy [4] (unless you allow random noise [3]).

Therefore, although it may well be that Backgammon is inherently well-suited to co-evolutionary learning [23], that does not fully explain why co-evolution performs well on some tasks that are unlike Backgammon (such as IPD), or why co-evolution on any task stagnates at the level it does.

2.1.3 This Paper Examines Initial Diversity

This paper looks at the following possibility. Given that, in co-evolutionary learning, the learning environment is the current population, then increasing the genetic diversity in the population might add variety to the learning environment, thus avoid stagnation for longer, and so rise to a higher level of ability. Perhaps that would explain why co-evolution works well on IPD, despite its lack of random dice.

It would seem reasonable to expect that more genetic diversity in the initial population would give rise to a more varied learning environment. Section 4.2 demonstrates that this appears to be the case, but unfortunately Section 4.3 will show that the true situation is more complicated.

2.2 Iterated Prisoner's Dilemma

Iterated Prisoner's Dilemma (IPD) is an abstract mathematical game which is widely studied in political science, biology, and artificial intelligence [1]. In its basic form, the Prisoner's Dilemma is a two-player game where each player has two choices, and the payoff matrix for those two choices (Figure 2) satisfies the following conditions [1]:

- $T > R$ and $P > S$ (Defection always pays more)
- $R > P$ (Mutual cooperation beats mutual defection)
- $R > (S + T)/2$ (Alternating doesn't pay)

Many choices of parameters satisfy the conditions of an IPD shown in Figure 2. This paper uses the values $R = 4$, $T = 5$, $S = 0$, and $P = 1$, as shown in Figure 3.

In *iterated* prisoner's dilemma, the game in Figure 2 is played not just once, but many times, with a memory of previous iterations. This allows the possibility of retaliation and mutual cooperation.

The iterated game is widely studied because it contains the basics of many real-world situations. Studies that use IPD to explain human behavior include the U. S. Watergate scandal of 1972-75 [22], the Cold War of 1945-1990 [5], and life in

	Cooperate	Defect
Cooperate	R	T
Defect	S	P

Figure 2: The payoff matrix for the 2-player prisoner’s dilemma game. The values S, P, R, T must satisfy $T > R > P > S$ and $R > (S + T)/2$.

	Cooperate	Defect
Cooperate	4	5
Defect	0	1

Figure 3: The payoff matrix for the 2-player prisoner’s dilemma game as used here.

the trenches of the First World War of 1914-1918 [1, pages 73-87].

Co-evolutionary learning applied to IPD demonstrates how sometimes mutual cooperation can dominate, even without any central authority to enforce cooperation [2]. Figure 5 shows a typical co-evolutionary run in which mutual cooperation eventually dominates.

This “evolution of cooperation” is occasionally punctuated by mass extinctions [9, page 287] [13, page 141] [19], which occur without any external cause (e.g., without meteor impacts, ice ages, or habitat destruction).

2.3 IPD with More Choices

In the classic version of IPD, each player only has two choices: full cooperation or full defection. In real-world situations, however, cooperation is rarely all-or-nothing [24, page 175]. Players can often choose among varying degrees of cooperation. For example, polluters can adjust the amount of pollution they generate, to avoid provoking complaints from nearby residents. To capture this realism, some recent papers add fine-grained intermediate levels of cooperation to the game of IPD [13] [24] [27].

In this paper, intermediate choices of cooperation are implemented as a straightforward generalization of the 2-choice case. For example, if each player has four choices (instead of

the usual two), the payoff matrix we use is in Figure 4, which shows the payoff to the player on the left, player A. Note that the four corners of Figure 4 are the same payoffs as for the two-choice game — it is merely interpolation.

		Player B			
		-1	$-\frac{1}{3}$	$+\frac{1}{3}$	+1
Player A	-1	1	$2\frac{1}{3}$	$3\frac{2}{3}$	5
	$-\frac{1}{3}$	$2\frac{2}{3}$	2	$3\frac{1}{3}$	$4\frac{2}{3}$
	$+\frac{1}{3}$	$1\frac{2}{3}$	3	$4\frac{1}{3}$	
	+1	0	$1\frac{1}{3}$	$2\frac{2}{3}$	4

Figure 4: The payoff matrix for the 2-player prisoner’s dilemma game, with four choices of cooperation level, where -1 is full defection and +1 is full cooperation. This shows the payoff to Player A.

In Figure 4, the payoff to player A is given by:

$$p_A = 2.5 - 0.5c_A + 2c_B, (-1 \leq c_A, c_B \leq 1) \quad (1)$$

Here c_A and c_B are the cooperation levels of the two players, which are evenly discretized into as many fine-grained choices of cooperation as desired.

This paper will mostly consider the case where each player has sixteen evenly discretized choices of cooperative level, giving a 16×16 payoff matrix.

Only an even-numbered number of choices are used, so that the line between two payoff levels is at the zero point of cooperation (equidistant between full cooperation and full defection). Otherwise it tends to drift around that central step level without finding its edge.

In the general case, the payoff values must satisfy the following three conditions (Equations 2 through 4) to be a Prisoner’s Dilemma game, even with intermediate values of cooperation. These conditions are a generalization of the two-choice conditions in Section 2.2. Here, c_A and c_B are the cooperative levels of two players A and B, and p_A is the payoff to player A.

- Defection always pays more. For $c_A < c'_A$ and constant c_B :

$$p_A(c_A, c_B) > p_A(c'_A, c_B) \quad (2)$$

- Mutual cooperation beats mutual defection. For $c_B < c'_B$ and constant c_A :

$$p_A(c_A, c_B) < p_A(c_A, c'_B) \quad (3)$$

- Alternating doesn’t pay. For $c_A < c'_A$ and $c_B < c'_B$:

$$p_A(c'_A, c'_B) > \frac{1}{2} (p_A(c_A, c'_B) + p_A(c'_A, c_B)) \quad (4)$$

3 Genetic Algorithm Implementation Details

Here, a genetic algorithm (GA) maintains a population of 100 trial strategies for 2-player IPD. Every generation of the GA, each strategy plays against other members of the current population (including itself). A player's fitness is its average score over all this generation's games. If a player makes it to the next generation with no change, it still gets a new fitness by playing all the members of that new generation.

In each game of IPD between two players, each player can see the opponent's recent actions, and their own recent actions. In the runs shown here, this remembered history is limited to only the most recent iteration, i.e., the memory of the past is only 1 iteration. This is sufficient to allow for retaliation, and learning strategies like "Tit-for-tat" [1].

To start a game, each player has in its genotype what it presumes is the pre-game "history", that affects its first move. In effect, the first move is wired into the genotype.

3.1 The Shadow of the Future

In IPD, the "shadow of the future" [1, page 13] is an issue. If players could count, and if the number of iterations in a game of IPD were fixed, then players would have no incentive to cooperate on the very last move, because there is no risk of retaliation. But if every player thus plans to defect on the very last move, then they would also have no incentive to cooperate on the *second* last move (seeing as nobody will cooperate on the last move anyway). And so on back to the start of the game. Thus, cooperation only emerges if the game length is somehow kept uncertain.

One way to keep game length uncertain is to have a fixed probability of ending the game on each successive move [2]. A given value of this probability will keep average game length about the same, while keeping the exact game length uncertain (to encourage cooperation).

However, in this paper, individual strategies are represented in such a simple way (feedforward neural networks) that they cannot count. This lets us use a fixed-length game, with no chance of the players figuring out that games last for a predictable number of iterations. In this paper, game length is 150 iterations, comparable to the average length used by Axelrod [2].

3.2 Representation of Individual Strategies

Each member of the population is a fixed-length string of floating-point numbers. These represent the weights and biases of a feed-forward neural network with 20 hidden nodes, and one output node. For the one remembered previous iteration of IPD, there are four inputs.

1. One for one's own earlier level of cooperation.
2. One for the opponent's earlier level of cooperation.
3. An input which is 1 if the opponent exploited the player, and zero otherwise.

4. An input which is 1 if the opponent was exploited by the player, and zero otherwise.

For the last two inputs, "exploited" means that the players' cooperation levels differ, so that the player with lower cooperation receives a higher payoff. Even if payoffs are different by a tiny amount, the corresponding input will still be 1. With this implementation, players need not learn to subtract, and can know immediately if they are being exploited (even by a small amount), which is important for fine-grained differences in the level of cooperation.

Each individual is a neural network with one output node. Like all the hidden nodes, that output node is sigmoided, to put its value in the range $[-1, +1]$.

After sigmoiding, the output is still continuous. This output is discretized in such a way that all the values are equidistant from each other. For example, if there were 4 choices of cooperation, the output node's value in the range $[-1, +1]$ would be adjusted to the nearest of the values $(-1, -\frac{1}{3}, +\frac{1}{3}, +1)$. This discretized output is the player's choice of cooperation for the current move. As mentioned in Section 2.3, this paper will use 16 discrete levels of cooperation.

3.3 Other Genetic Algorithm Parameters

This work used PGA-Pack software, courtesy of the Argonne National Laboratory. The settings used here are entirely conventional, and include the following.

Mutation is Gaussian, with a standard deviation of 0.15. The probability of mutating a particular allele (a weight or bias in the feedforward neural network) is 0.2.

Crossover is uniform, with a 0.25 probability of an offspring being crossed over. An offspring suffers either mutation or crossover, but not both, so there is a 0.75 chance of being mutated.

Elitism is used: the best single strategy is copied unchanged to the next generation (where it gets re-evaluated by playing all the members of the new generation — the old fitness value does not get copied).

Selection was stochastic universal selection with linear ranking: the best player in the population could expect two offspring, and the worst could expect zero offspring.

4 Results

4.1 Evolution of Cooperation: 2 Levels of Cooperation

With a population of 100, IPD games of length 150 iterations, and with only 2 choices for cooperation (full cooperate or full defect), Figure 5 shows the classic, Axelrod-style evolution of cooperation [2]. There is an initial dip, but then mutual cooperation dominates. Cooperation maintains its dominance, except for the occasional mass extinction [9, page 287] [13, page 141] [19].

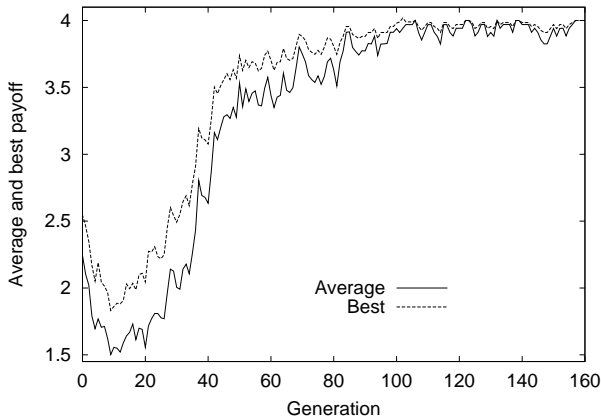


Figure 5: This shows the “evolution of cooperation” [2] in a typical co-evolving population that plays IPD with only 2 choices of cooperation level. After an initial dip, average payoff rises to full cooperation and stays there, apart from a rare mass extinction [9, page 287] [13, page 141] [19].

4.2 A Difficult Learning Task: 16 Levels of Cooperation

Many real-world problems that resemble the IPD offer a fine-grained choice of responses: for example, polluters can fine-tune the amount of pollution on a daily basis. Thus, there is great interest in how to encourage cooperation in the fine-grained version of IPD, to give insights into how to promote cooperation in similar real-world conflicts.

With only two stark choices, mutual cooperation often dominates — it is a much simpler situation. But increasing the number of available choices (i.e., making the range more fine-grained) tends to make cooperation less likely to dominate [27, Section 4.3]. The IPD game with a range of choices is a qualitatively different contest, and offers more scope for exploiting opponents.

This section checks if genetic diversity in the initial population might encourage cooperation in IPD with 16 choices of cooperation, to perhaps gain insight into solving real-world conflicts with shades of cooperation. To do so, this section varies the range of values for the initial population, to observe whether cooperation will subsequently dominate. There are no limits on values for subsequent generations, and mutation is free to exceed these initial limits.

Figure 6 shows the population’s average payoff for runs whose initial population has values uniformly random in the interval $(-0.8, +0.8)$. That is, initial genetic diversity is low. Only the first randomized population has its values limited to $(-0.8, +0.8)$, and in subsequent generations mutation is free to exceed these values.

Note that in almost all of the runs in Figure 6, mutual defection dominates. Cooperation hardly ever takes hold. With low initial diversity, cooperation is unlikely.

Figure 7 shows the population’s average payoff for runs whose initial population has values uniformly random in the interval $(-1.25, +1.25)$. Again, subsequent generations are

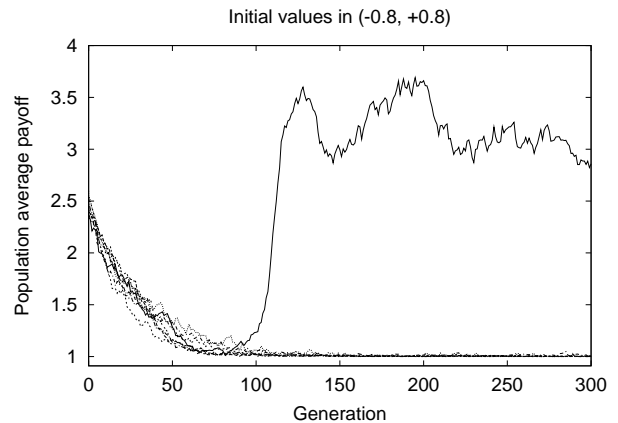


Figure 6: Ten co-evolutionary runs of IPD with 16 choices. Initial genetic diversity is low, in the range $[-0.8, +0.8]$. Low-scoring strategies that defect dominate in almost all runs.

free to exceed these values. With this greater initial diversity, mutual defection dominates in about half the runs in Figure 7, with mutual cooperation (to some degree) in the other half of the runs.

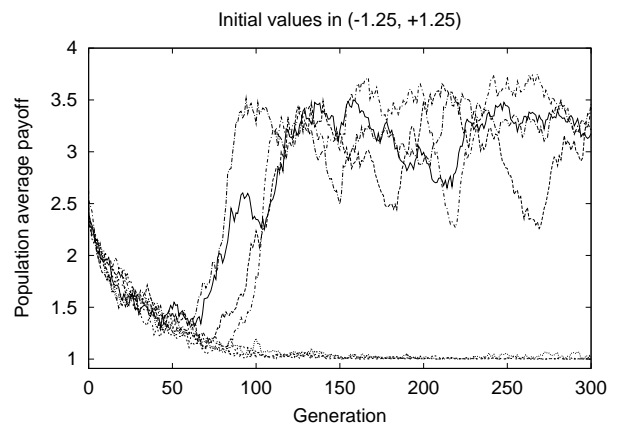


Figure 7: Ten runs of IPD with 16 choices, with moderate initial genetic diversity. About half discover some degree of mutual cooperation.

Figure 8 shows the population’s average payoff for runs whose initial population has values uniformly random in the interval $(-1.5, +1.5)$. Again, later generations may exceed these values. With this healthy initial genetic diversity, mutual cooperation eventually dominates in almost every run in Figure 8.

In Figures 6 through 8, the only difference is in the range of random values for the initial population. Values in subsequent generations do not have limits. Yet varying the initial diversity makes a huge difference in eventual outcome, i.e., whether or not mutual cooperation dominates.

It seems plausible that the relationship is causal, i.e., that higher diversity encourages cooperation. As discussed in Section 1.2, the learning environment in co-evolution is, in

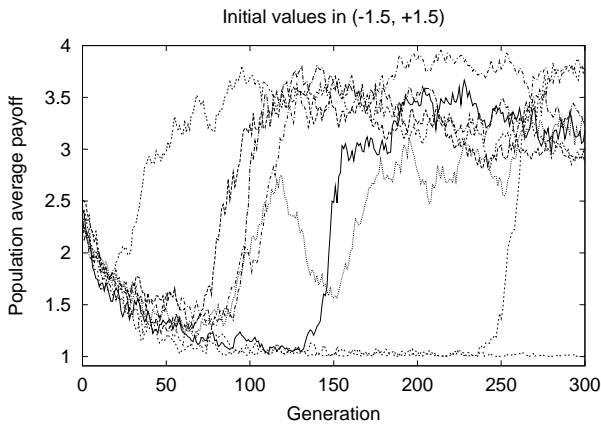


Figure 8: Ten runs of 16-choice IPD, with high initial genetic diversity. Almost all discover some degree of mutual cooperation.

fact, the other members of the evolving population; so a more varied population should be a more varied learning environment, which might cause better and more robust learning.

So why not end the paper now, on an optimistic conclusion? It looks like greater diversity causes better learning, and in the game of IPD promotes cooperation as well— what a happy ending that would be.

Instead, Section 4.3 will show that things are not as they seem, and that increasing initial genetic diversity is, in fact, perverting the learning process.

4.3 The Real Cause: Problem-Specific Side Effects

Section 4.2 appears to support the notion that more *genetic* diversity encourages mutual cooperation — a rosy and comforting possibility.

This section looks at exactly what the initial population is doing. Given that players can choose from 16 discrete levels of cooperation, what parts of that 16×16 payoff matrix does the initial population explore? That is, what is the *behavioral* diversity of the initial population, its phenotypic diversity?

Figure 4 shows a payoff matrix for only 4 choices of cooperation (as all 16 is an unreadable mess). For 16 choices of cooperation (as used here) the 16×16 payoff matrix still has the same scores in its corners as in the simpler 2×2 payoff matrix of Figure 3.

Figures 9 to 11 show the distribution of outcomes in the 16×16 payoff matrix, as chosen by a random initial population as it plays itself. These next few figures are histograms; the higher above the flat 16×16 base, the more often that particular outcome of the 16×16 payoff matrix was chosen by members of the random initial population. The z -axis for Figures 9 to 11 is the same scale for all three figures.

Incidentally, these next few figures will have some symmetry about the $x - y$ plane's diagonal, because it's arbitrary which of two players is Player 1 and Player 2 — that diagonal symmetry is merely an artifact.

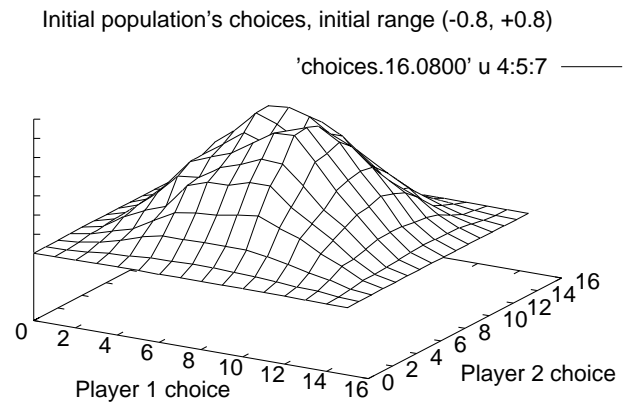


Figure 9: Low initial diversity in population, in the range $(-0.8, +0.8)$, explores only the center region of the 16×16 payoff matrix.

Figure 9 shows the choices of a typical initial population, whose values are uniformly random in the range $(-0.8, +0.8)$. It only explores payoff values in the center of the 16×16 payoff matrix. Players mostly choose middle levels of cooperation, ignoring the extremes of full cooperation or full defection — it is these intermediate values that make IPD more challenging than the classic two-choice version.

In Figure 9, the low genetic diversity means there is also low behavioral diversity — it does not evenly sample all possible outcomes of the 16×16 payoff matrix, only those in the middle.

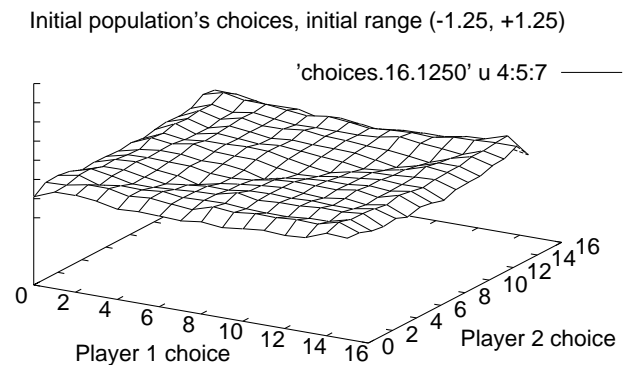


Figure 10: Modest initial diversity in the population, when initialized in the range $(-1.25, +1.25)$, evenly explores the 16×16 payoff matrix. The diagonal symmetry is an artifact.

Figure 10 shows the choices of the initial population with uniform values in the range $(-1.25, +1.25)$. It has a much smoother sampling over the whole of the 16×16 payoff matrix, i.e., Figure 10 has more behavioral diversity.

Figure 11 shows the choices of an initial population with values in the range $(-1.5, +1.5)$. This bucks the trend of genetic diversity being a good thing in this learning task. Instead of evenly sampling the whole of the 16×16 payoff matrix,

the four corners are being heavily sampled, and the outcomes with intermediate levels of cooperation (i.e., in the middle of the 16×16 payoff matrix) are essentially ignored.

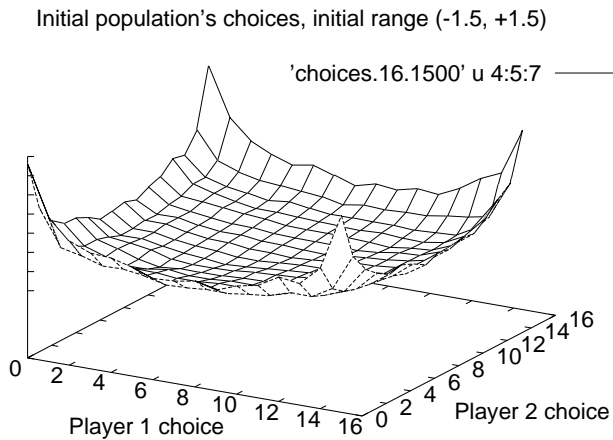


Figure 11: High initial diversity in the population, when initialized in the range (-1.5, +1.5), only explores the corners of the 16×16 payoff matrix.

The intermediate levels between full cooperation and full defection are virtually ignored in Figure 11. But it is precisely those intermediate levels that make the 16×16 game more challenging than the simpler 2×2 game — in the simpler 2×2 game, cooperation is more likely.

What's happening in Figure 11 is that the more difficult intermediate choices have been avoided by increasing the range of the initial population. No wonder cooperation always emerges in Figure 8.

This demonstrates that greater *genetic* diversity does not necessarily cause *behavioral* diversity or phenotypic diversity. In this case, it's the opposite — beyond a certain point, a larger range of values causes *less* diverse behavior, so that only a fraction of the 16×16 payoff matrix is being explored.

5 Discussion

5.1 Intermediate Choices Make Cooperation Harder

Recall from Section 4.1 that in IPD with only two stark choices, cooperation is likely to dominate [2]. Only two choices gives a simpler game, with a greater tendency for mutual cooperation to dominate, as it did in Figure 5.

In contrast, allowing intermediate degrees of cooperation in IPD makes it less likely for high-scoring cooperative strategies to dominate [13] [27]. That is, the extra complexity of those additional choices makes it more difficult for a learning algorithm to achieve high-scoring strategies. The game with sixteen choices between full cooperation and defection (with its 16×16 payoff matrix) is less likely to experience the “evolution of cooperation” [27, Section 4.3].

In the real world, the existence of intermediate levels of cooperation makes cooperation so much less likely, that in

many situations those intermediate levels are made unavailable. For example, the 1972 Anti-Ballistic Missile Treaty [6] outlaws certain weapons that can reduce the severity of a nuclear war². This might seem perverse at first glance, but by removing any intermediate levels between total non-use of nuclear weapons and full-scale war, the treaty (supposedly) makes total war less likely.

However, most real-world situations do not offer an authority who can abolish those complicated (and conflict-causing) intermediate degrees of cooperation. The most interesting situations are precisely those where a central authority does not exist, but cooperation is nonetheless desirable, such as in international conflicts [5].

If players were to agree to abstain from fine-grained intermediate levels of cooperation, as they do for the Anti-Ballistic Missile Treaty [6], then it makes cooperation more likely. But doing so avoids the possibility of learning the more complicated game that offers those intermediate values.

5.2 Higher Genetic Diversity Avoids Intermediate Choices

Figure 8 says the highest initial diversity (in the range (-1.5, +1.5)) almost always enjoyed mutual cooperation. The reason is clear from Figure 11: the initial population is not really playing on a 16×16 payoff matrix at all — it's basically only considering the simpler 2×2 game (in which cooperation is more likely to emerge [27, Section 4.3]) and ignoring the intermediate values of cooperative level. No wonder mutual cooperation occurs more often.

Figure 11 isn't learning the more complicated 16×16 game at all — it's only learning the simpler 2×2 game and avoiding the extra complications of the larger game.

For this learning task, genetic diversity is a red herring, and is *not* the main cause of the cooperation in Figure 8. It merely manages to avoid the tricky intermediate choices, just as the 1972 Anti-Ballistic Missile Treaty [6] makes nuclear war less likely by preventing partial nuclear war.

Section 4.3 showed that higher genetic diversity actually reduced the behavioral diversity; instead of doing actions throughout the 16×16 payoff matrix, the highest genetic diversity skewed the range of decisions away from the challenging intermediate values of fine-grained choices, and towards the much simpler 2×2 game.

5.2.1 Diversity and Other Learning Tasks

Of course, the results of this one particular learning task do not necessarily generalize to all learning tasks. For many learning tasks, more genetic diversity may be beneficial. However, Section 4.2 has shown that merely demonstrating

²Specifically, the treaty outlaws defenses against incoming nuclear missiles, such as the Strategic Defense Initiative (Star Wars). It allows one local-scale system on each side: one protects Moscow, the other protects a missile field near Grand Forks, North Dakota [6]. This treaty is a cornerstone of the controversial doctrine of Mutually Assured Destruction (MAD). MAD argues that the use of nuclear weapons can be prevented by guaranteeing that any nuclear attack, no matter how limited, will cause all-out nuclear war.

a correlation between genetic diversity and some effect (in this case, more cooperation) can be misleading.

It is not enough to demonstrate empirically that higher genetic diversity merely *correlates* with some other measurable quantity. What is needed is to demonstrate causality. If this paper had ended in Section 4.2, it would have given a plausible but erroneous conclusion.

Such an incorrect conclusion would make this paper resemble the airline passenger who asked the pilot to stop turning on the “Fasten seat belts” sign, believing the sign caused the turbulence and not the other way around — another correlation that is not causal.

6 Conclusions

Genetic diversity is often believed to cause a multitude of effects in evolutionary computation. As a result, methods to increase or maintain diversity receive study: mutation operators [16], population initialization [12], and even pseudo-random number generators [21] have all been examined in this context. This paper looks at initial population diversity.

The learning task, the game of Iterated Prisoner’s Dilemma (IPD), used additional fine-grained intermediate levels of cooperation, instead of the classic version’s two choices of only total cooperation or total defection. Fine-grained choices more closely resembles a variety of real-world situations, and is a harder game in that cooperation is less likely to occur [27, Section 4.3].

Section 4.2 appeared to show that more initial genetic diversity made it more likely that mutual cooperation would dominate. Unfortunately, closer inspection in Section 4.3 demonstrated that this was not the case: the higher genetic diversity was merely reducing the behaviorally diversity of the players, so that they were effectively playing the simpler IPD game with only two choices of cooperative level. That is, higher genetic diversity did not encourage the emergence of cooperation, it merely avoided the complexity of IPD with more choices.

This demonstrates how easy it can be to mistakenly blame genetic diversity (or its lack) for the performance of evolutionary computation. One must find the cause of performance issues, rather than merely show correlation.

Acknowledgments

The authors are grateful for discussions with members members of the University of Queensland’s Cognitive Science Group, and the Computational Intelligence Group at the Australian Defence Force Academy. This work used PGA-Pack software, courtesy of David Levine at the Argonne National Laboratory. Gerald Tesauro of IBM’s Thomas J. Watson Research Center made available the Backgammon benchmark strategy Pubeval. Paul Darwen’s research is supported by the Australian Research Council (ARC).

Bibliography

- [1] Robert M. Axelrod. *The Evolution of Cooperation*. Basic Books, New York, 1984.
- [2] Robert M. Axelrod. The evolution of strategies in the iterated prisoner’s dilemma. In Lawrence Davis, editor, *Genetic Algorithms and Simulated Annealing*, chapter 3, pages 32–41. Morgan Kaufmann, Los Altos, California, 1987.
- [3] Robert Boyd. Mistakes allow evolutionary stability in the repeated prisoner’s dilemma game. *Journal of Theoretical Biology*, 136:47–56, 1989.
- [4] Robert Boyd and Jeffrey P. Lorberbaum. No pure strategy is evolutionarily stable in the repeated prisoner’s dilemma game. *Nature*, 327:58–59, 7 May 1987.
- [5] Steven J. Brams. *Superpower Games*. Yale University Press, 1985.
- [6] Richard Dean Burns. *Encyclopedia of arms control and disarmament*. Maxwell Macmillan, Toronto, 1993.
- [7] Kumar Chellapilla and David B. Fogel. Evolution, neural networks, games, and intelligence. *Proceedings of the IEEE*, 87(9):1471–1496, September 1999.
- [8] Paul J. Darwen and Jordan B. Pollack. Co-evolutionary learning on noisy tasks. In *Congress on Evolutionary Computation*, pages 1724–1731. IEEE Press, July 1999.
- [9] Paul J. Darwen and Xin Yao. On evolving robust strategies for iterated prisoner’s dilemma. In *Progress in Evolutionary Computation*, volume 956 of *Lecture Notes in Artificial Intelligence*, pages 276–292. Springer, Berlin, 1995.
- [10] Paul J. Darwen and Xin Yao. Speciation as automatic categorical modularization. *IEEE Transactions on Evolutionary Computation*, 1(2):101–108, July 1997.
- [11] David Gardner and Tom Gardner. *The Motley Fool Investment Guide*. Fireside, 1997.
- [12] John J. Grefenstette. Incorporating problem specific knowledge into genetic algorithms. In Lawrence Davis, editor, *Genetic Algorithms and Simulated Annealing*, chapter 4, pages 42–60. Morgan Kaufmann, San Mateo, California, 1987.
- [13] Paul G. Harrald and David B. Fogel. Evolving continuous behaviors in the Iterated Prisoner’s Dilemma. *Biosystems*, 37:135–145, 1996.
- [14] W. Daniel Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In *Artificial Life 2*, pages 313–323. Addison-Wesley, Redwood City, California, 1991.

- [15] Philip Husbands and Frank Mill. Simulated co-evolution as the mechanism for emergent planning and scheduling. In Richard K. Belew and Lashon B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 264–270. Morgan Kaufmann, July 1991.
- [16] Cornelia Kappler. Are evolutionary algorithms improved by large mutations? In *Parallel Problem Solving from Nature – PPSN4*, volume 1141 of *Lecture Notes in Computer Science*, pages 346–355, Berlin, September 1996. Springer.
- [17] John R. Koza. *Genetic Programming: on the programming of computers by means of natural selection*. MIT Press, Cambridge, Massachusetts, 1992.
- [18] Wei-Po Lee, John Hallam, and Henrik H. Lund. A hybrid gp/ga approach to co-evolving controllers and robot bodies to achieve fitness-specific tasks. In *Proceedings of the 1996 IEEE Conference on Evolutionary Computation*, pages 384–389, Nagoya, Japan, May 1996. IEEE Press.
- [19] Kristian Lindgren. Evolutionary phenomena in simple dynamics. In *Artificial Life 2*, pages 295–312. Addison-Wesley, Redwood City, California, 1991.
- [20] Burton Gordon Malkiel. *A Random Walk down Wall Street*. Norton, New York, seventh edition, 1999.
- [21] Mark M. Meysenburg and James A. Foster. The quality of pseudo-random number generators and simple genetic algorithm performance. In *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 276–281. Morgan Kaufmann, 1997.
- [22] Douglas Muzzio. *Watergate Games: strategies, choices, outcomes*. New York University Press, 1982.
- [23] Jordan B. Pollack and Alan D. Blair. Co-evolution in the successful learning of Backgammon strategy. *Machine Learning*, 32(3):225–240, 1998.
- [24] Gilbert Roberts and Thomas N. Sherratt. Development of cooperative relationships through increasing investment. *Nature*, 394:175–179, 9 July 1998.
- [25] Gerald Tesauro. Comments on “Co-evolution in the successful learning of Backgammon strategy”. *Machine Learning*, 32(3):241–243, 1998.
- [26] Xin Yao and Paul J. Darwen. An experimental study of N-person iterated prisoner’s dilemma games. *Informatika*, 18:435–450, 1994.
- [27] Xin Yao and Paul J. Darwen. How important is your reputation in a multi-agent environment. In *Proceedings of the 1999 IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 575–580, Tokyo, October 1999.